5
Resultados e Aplicações

Os principais resultados desta pesquisa são apresentados neste capítulo, divididos em duas partes. A Seção 5.1 aborda a geração de ADFs utilizando a representação proposta nos Capítulos 3 e 4. Em seguida, a Seção 5.2 apresenta resultados referentes ao uso das ADFs em aplicações de visualização aceleradas por GPU.

5.1
Construção de ADFs

A construção de ADFs é um processo caro, que pode levar de alguns minutos até várias horas. Neste trabalho, as ADFs são construídas por uma ferramenta auxiliar que recebe como entrada os seguintes parâmetros:

1. **Tipo:** a ADF construída pode ser *superficial* ou *global*. No primeiro caso, a iso-superfície representada é sempre a de iso-valor zero.

2. **Estratégia de construção:** pode ser *bottom-up* ou ladrilhada (vide Seção 2.2.3). Em alguns casos, a estratégia *bottom-up* é mais eficiente; porém, ela não escala para ADFs acima do nível 9.

3. **Nível máximo:** especifica o tamanho do volume usado na construção *bottom-up*; ou o nível máximo até onde uma célula pode ser subdividida, na construção ladrilhada.

4. **Erro máximo:** especifica uma tolerância para o erro de reconstrução das células. O processo de subdivisão da ADF é interrompido nas células onde o erro estimado é menor que o erro máximo tolerado.

5. **Função de distância:** define o *campo de distância com sinal* a ser representado pela ADF. Atualmente, as opções são:

   (a) Especificar uma função de distância primitiva; *e.g.* esfera ou torus.

   (b) Especificar um arquivo no formato PLY, OFF, 3DS ou OBJ, que contenha uma malha de triângulos fechada. As técnicas envolvidas no cálculo do campo de distância para malhas de triângulos são resumidas na Seção 5.1.1.

Com base nos parâmetros fornecidos, a ferramenta constrói a ADF em cinco estágios, na seguinte ordem:

1. Construção da ADF a partir da função de distância fornecida.
2. Cálculo dos gradientes.
3. Geração da textura de normais suaves.
4. Geração da textura de *voxels*.
5. Geração da função de dispersão perfeita para a *octree*.

5.1.1 Distância para Malhas de Triângulos

Toda malha de triângulos fechada e orientável pode definir um campo de distância com sinal. A questão é como calcular o campo de distância de forma eficiente. Este problema pode ser dividido em duas partes: o cálculo das distâncias (sem sinal), e o cálculo dos sinais. Para o cálculo das distâncias, existem duas abordagens básicas:

1. Usar subdivisão espacial hierárquica para acelerar o descarte de triângulos que não contêm o ponto mais próximo. Neste caso, as distâncias calculadas são *exatas*. Em geral, para um modelo com n triângulos, a construção da estrutura espacial tem custo $O(n \log n)$, e cada consulta tem custo $O(\log n)$.
2. Usar um volume com uma transformada de distância. Neste caso, as distâncias são *aproximações*, cuja precisão é limitada à resolução do volume. Para um volume com m células, o cálculo da transformada tem custo $O(n + m)$, e cada consulta tem custo $O(1)$.

Neste trabalho, foram implementadas três técnicas para o cálculo de distâncias:

1. Cálculo exato, baseado em uma árvore de caixas envolventes orientadas (*OBB-tree*), utilizando o método descrito no trabalho [16].
2. Cálculo exato, baseado em uma árvore k-dimensional (*kD-tree*), utilizando um método próprio. Esta técnica utiliza menos memória que a árvore de caixa orientadas, mas é ligeiramente mais lenta.
3. Cálculo aproximado, baseado na transformada de distância proposta no trabalho [29]. Esta técnica é muito rápida, mas só funciona bem com a construção *bottom-up*; portanto, não escala para ADFs acima do nível 9.

Os sinais do campo de distância são sempre determinados através da técnica de pseudo-normais proposta no trabalho [1].

5.1.2
Resultados para a Construção de ADFs

A Tabela 5.1 apresenta resultados quantitativos para as ADFs retratadas nesta dissertação. As três primeiras colunas contêm o nome do modelo, a figura em que ele aparece e o número de triângulos da malha, respectivamente. As três colunas seguintes — tipo, nível máximo e erro máximo — indicam os parâmetros usados na construção da ADF. Finalmente, as três últimas colunas indicam o número de células armazenadas na ADF, as dimensões da textura de *voxels* e o tempo total de construção da ADF.

Modelo	Fig.	# Triâng.	Tipo	Nível Máx.	Erro Máx.	# Células	Dimensões Tex. *Voxels*	Tempo
Eros	4.1	394K	global	8	$3e^{-4}$	187,297	$171^2 \times 174$	13m 10s
			superficial	9	$1e^{-4}$	400,997	222^3	7m 59s
Estátua Tailandesa	5.1	10M	global	8	$3e^{-4}$	187,448	$171^2 \times 174$	5h 57m 22s
			superficial	9	$1e^{-4}$	319,669	204×207^2	12m 18s
Armadillo	5.2	346K	global	8	$3e^{-4}$	489,315	237^3	9m 22s
Ramsés	5.3	387K	global	8	$3e^{-4}$	246,661	189^3	11m 26s
			superficial	9	$1e^{-4}$	214,712	180^3	7m 8s
Caixa Circular	5.4	361K	global	8	$3e^{-4}$	740,718	270×273^2	12m 54s
Mão da Magali	5.4	397K	global	8	$3e^{-4}$	235,906	186^3	18m 21s
			superficial	9	$1e^{-4}$	282,673	$198^2 \times 195$	7m 11s

Tabela 5.1: Resultados quantitativos para ADFs retratadas nesta dissertação.

Para efeito de comparação, todas as ADFs foram construídas pela estratégia *bottom-up*. O cálculo do campo de distância das ADFs superficiais foi feito por transformadas de distância, e o das ADFs globais foi feito por árvores de caixas envolventes orientadas.

5.2
Aplicações

As seções seguintes demonstram algumas aplicações baseadas em GPU que renderizam ADFs através das técnicas propostas neste trabalho. Todos os resultados foram obtidos em uma resolução de 512^2, utilizando OpenGL 2.1 e uma NVIDIA GeForce 8800 GTX 768MB.

5.2.1
Renderização de Modelos Sólidos

Uma das aplicações mais simples para o arcabouço proposto nesta dissertação é a renderização de *objetos sólidos*, representados implicitamente pelas iso-superfícies de valor zero das ADFs. Como exemplo, a Figura 5.1 mostra a Estátua Tailandesa de Stanford renderizada na GPU, com normais suaves, a partir de uma ADF superficial de nível 9.

Figura 5.1: Estátua Tailandesa renderizada em GPU com normais suaves.

A última coluna da Tabela 5.2 contém as taxas de renderização obtidas para as mesmas ADFs da Tabela 5.1. A taxa refere-se à média de quadros por segundo obtida quando as iso-superfícies de valor zero das ADFs são renderizadas, utilizando-se as técnicas descritas no Capítulo 4. Em todos os casos, foram utilizadas tolerâncias de 10^{-3} para as interseções, e um limite de 64 iterações para o traçado por esferas.

Modelo	Figura	Tipo	Nível Máximo	Erro Máximo	Quadros/ Segundo
Eros	4.1	global	8	$3e^{-4}$	76
		superficial	9	$1e^{-4}$	68
Estátua Tailandesa	5.1	global	8	$3e^{-4}$	108
		superficial	9	$1e^{-4}$	92
Armadillo	5.2	global	8	$3e^{-4}$	76
Ramsés	5.3	global	8	$3e^{-4}$	104
		superficial	9	$1e^{-4}$	89
Caixa Circular	5.4	global	8	$3e^{-4}$	71
Mão da Magali	5.4	global	8	$3e^{-4}$	95
		superficial	9	$1e^{-4}$	82

Tabela 5.2: Desempenho da renderização de ADFs em GPU.

5.2.2 Extração de Camadas (Offsets)

Volumes de distância são comumente utilizados para calcular *superfícies envoltórias* para objetos sólidos. Com o auxílio do campo de distância o cálculo torna-se trivial, pois uma superfície envoltória é simplesmente uma

iso-superfície do campo de distância (com iso-valor positivo). Deste modo, não é necessário se preocupar com inter-colisões ou com a aparagem (*trimming*) da superfície, como nos métodos baseados em malhas poligonais.

As ADFs globais provêem precisão suficiente para este tipo de aplicação. Além disso, o algoritmo de lançamento de raios descrito no Capítulo 4 pode renderizar trivialmente qualquer iso-superfície da ADF — bastando, para tal, que o valor da variável c (o iso-valor) seja alterado. A Figura 5.2 demonstra o efeito causado por esta alteração para iso-valores negativos e positivos.

5.2(a): Iso-valor Negativo 5.2(b): Zero 5.2(c): Iso-valor Positivo

Figura 5.2: Iso-superfícies do modelo Armadillo ("tatu") de Stanford.

5.2.3 Níveis de Detalhes

Um modo muito simples para tratar níveis de detalhes, sugerido por Frisken *et al.* [10], consiste em truncar as ADFs em um nível fixo da *octree*. A Figura 5.3 ilustra o efeito obtido ao truncar-se uma ADF de nível 8 em quatro níveis diferentes da *octree*, durante a renderização. Observe que, como as ADFs são renderizadas por lançamento de raios, mudanças no nível de detalhe não afetam significativamente o desempenho da visualização.

Com esta abordagem simples, as transições entre níveis de detalhes são descontínuas e visualmente distrativas, a menos que os níveis envolvidos sejam muito refinados (maiores que 9). Para obter transições mais suaves, Frisken *et al.* propõem [10] um método alternativo, onde os níveis da ADF são truncados com base no erro de reconstrução das células.

5.2.4 Metamorfoses

Campos de distância também são comumente utilizados para simular metamorfoses entre objetos sólidos. Comparadas às técnicas baseadas em

5.3(a): Nível 5 5.3(b): Nível 6 5.3(c): Nível 7 5.3(d): Nível 8

Figura 5.3: Ramsés em quatro níveis de detalhes.

malhas poligonais, as técnicas de metamorfose volumétricas são mais fáceis de implementar e funcionam naturalmente para superfícies de gêneros distintos.

Uma aplicação de metamorfose muito simples foi implementada neste trabalho com base na técnica sugerida por Payne e Toga [23]. A aplicação carrega duas ADFs globais simultaneamente em GPU e utiliza uma variação do lançador de raios proposto no Capítulo 4 para renderizar uma *interpolação linear* entre os campos de distância. As normais suaves para a metamorfose são obtidas trivialmente, interpolando-se linearmente as normais suaves de ambas as ADFs. A Figura 5.4 ilustra o resultado de uma seqüência de metamorfose entre o modelo "Caixa Circular" e o modelo "Mão da Magali".

Infelizmente, na maioria dos casos, uma simples interpolação não é suficiente para simular metamorfoses convincentes. A razão disto é que, por si só, a interpolação não preserva e nem busca mesclar as características correspondentes dos objetos. Para melhorar a qualidade das metamorfoses, uma opção seria introduzir uma função de deformação (*warp*) para guiar a interpolação dos campos de distância, como proposto por Cohen-Or *et al.* [4].

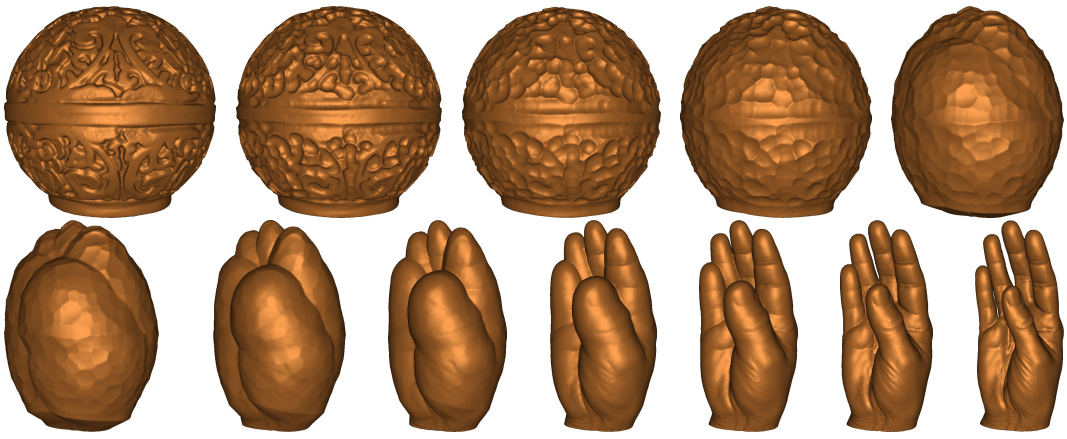

Figura 5.4: Interpolação entre ADFs, renderizada interativamente em GPU.